COMPUTATIONAL TINKERING

Programmare, risolvere, integrare

Kit C



il LABORATORIO della CURIOSITÀ

RICONNESSIONI
educazione al futuro

Xkè? Il laboratorio della curiosità
Via Gaudenzio Ferrari 1 - Torino
Tel. 011 81.29.786

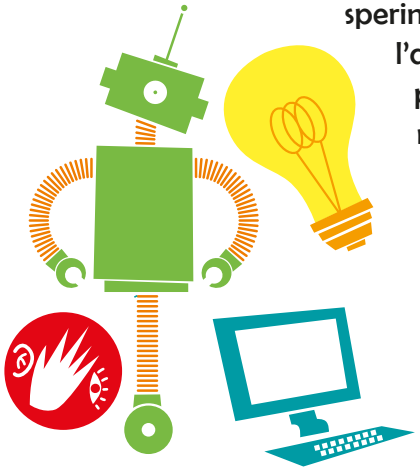
Sito web: **www.laboratoriocuriosita.it**
e-mail: **laboratoriocuriosita@xkezerotredici.it**

Riconnessioni
Piazza Bernini 5 - Torino
Tel. 011 43.06.511

Sito web: **www.riconnessioni.it**
e-mail: **riconnessioni@fondazionescuola.it**

Introduzione

Portare il *tinkering* a scuola significa lasciare spazio ad una modalità di apprendimento open *ended*, stimolare gli studenti a sviluppare la propria creatività scientifica, la capacità di sperimentare e "imparare facendo", procedendo per tentativi. Significa porre l'attenzione sul processo più che sul prodotto finale, sulla capacità di progettare, individualmente o in gruppo, soluzioni a problemi di varia natura e a metterle in pratica. Significa, ancora, valorizzare l'inventiva e il pensiero divergente, migliorare la capacità di concentrazione, la perseveranza, la gestione dell'incertezza, l'autostima e il pensiero critico.



Ripercorriamo le tappe del percorso proposto finora: il kit A vi ha permesso di realizzare progetti di *tinkering* "base", impiegando materiali e oggetti di uso comune (carta e cartone, plastica, metalli, ecc.), connettori (spago, elastici, nastro adesivo, ecc.) e semplici attrezzi. Con il kit B è stato possibile implementare i progetti con il movimento, la luce e il suono, introducendo alcuni componenti e semplici dispositivi elettrici (cavi, batterie, interruttori, lampadine, motorini, *buzzer* sonori).

Si vuole ora suggerire un altro passo, arrivando ad integrare le attività di *tinkering* con elementi abitualmente legati al mondo della **robotica**, del **coding** e delle **competenze digitali**. Il kit C è suddiviso in due parti distinte, in seguito presentate più dettagliatamente: il kit C1, proposto a tutte le classi della scuola primaria, che mette a disposizione dei robot educativi di piccole dimensioni (*Ozobot*); il kit C2, destinato alle classi terze, quarte e quinte, costituito da piccole schede programmabili (*micro:bit*).

In modo analogo a quanto proposto per il kit B, l'idea è costruire una "**library**" di robot educativi e materiali per il *coding* a disposizione di insegnanti e studenti: anche in questo caso, avrete dunque un periodo di tempo prestabilito per utilizzare il kit che avete ritirato. Al termine, il kit andrà riconsegnato e verrà messo a disposizione di altre classi. Potrete eventualmente usufruirne di nuovo in futuro! Per consentire agli utenti successivi di sfruttare lo stesso kit che avete ora tra le mani, vi invitiamo ad utilizzare e riordinare il materiale con cura.

Il kit C - Programmare, risolvere, integrare

Le esigenze poste dalla società in continuo cambiamento richiedono, anche al mondo della scuola, di preparare gli studenti a diventare "cittadini digitali". Questo non significa solamente insegnare le basi della programmazione e della robotica, ma anzi, in un contesto più ampio, sviluppare quello che viene definito il **pensiero computazionale**. Si tratta di aiutare gli studenti a rafforzare la dimensione logico-creativa, la capacità di elaborare un pensiero creativo per giungere alla soluzione di un dato problema. Il **coding** diventa uno strumento per tutti, per imparare a progettare, costruire (o "mettere in pratica"), decidere.

Dall'integrazione tra il *tinkering* e il pensiero computazionale, si giunge al **computational tinkering**! Entrambi gli approcci richiedono infatti di mettere in campo un mix di logica, razionalità, inventiva, intuito e capacità di comunicare con un linguaggio semplice ed efficace. Entrambi richiedono di sviluppare competenze di *problem posing* (individuare e comprendere il problema), *problem solving* (progettare e mettere in atto una soluzione adeguata al problema stesso) e la verifica, anche per tentativi successivi, che la strategia elaborata funzioni effettivamente.

Attraverso il *coding* è possibile allenarsi a scomporre problemi complessi in parti più piccole e gestibili, se affrontate una per volta. Si impara in tal modo a pensare su diversi livelli di astrazione, a suddividere il processo decisionale in singoli step, a ragionare passo passo sul modo migliore per ottenere un obiettivo e a pianificare una strategia come sequenza di azioni ordinate, ciascuna identificata da una specifica istruzione (*algoritmo*).

Tutte queste competenze possono ovviamente essere trasposte nella dimensione reale e nel *tinkering*: anche in questo caso, infatti, si parla di **processi**! Il *computational tinkering* dà concreta applicazione a questi concetti, permettendo di integrare i robot e le schede programmabili in manufatti e nuovi progetti realizzati con i materiali di uso più comune, forniti con i precedenti kit.

Pronti... via!

Si suggerisce di pianificare l'attività in classe con il kit C con un **minimo di 1 o 2 incontri** (rispettivamente per il kit C1 e per il kit C2), ciascuno della durata indicativa di 2 ore. Ovviamente, anche in questo caso l'età degli studenti costituirà un fattore di cui l'insegnante dovrà tenere necessariamente conto: l'attività potrà essere rimodulata in termini di tempi, numero di studenti in ciascun gruppo di lavoro e sfide proposte, al fine di "personalizzare" il percorso per la propria classe.

L'attività con il kit C potrà ricalcare, a grandi linee, le fasi di lavoro principali suggerite nella **guida** al *tinkering* fornita con i kit precedenti, a cui si rimanda per una trattazione più approfondita. Tenuto conto che gli studenti avranno già acquisito una certa familiarità con la metodologia, non sarà però necessario esplicitare la suddivisione dell'attività nelle varie fasi proposte, né dedicare un intero incontro a ciascuna di esse:

***Prima* - Accendere la creatività**
Organizzare gli spazi, stabilire poche e semplici regole condivise, "costruire" le condizioni al contorno che favoriscono il processo creativo.

***Tappa 1* - ESPLORAZIONE**
Scoprire e testare caratteristiche e funzionamento dei nuovi dispositivi. Nel caso del presente kit, questo significa ovviamente anche acquisire un minimo di familiarità con gli specifici linguaggi di programmazione: conoscerne i rudimenti consentirà di immaginare potenzialità e applicazioni e dunque di acquisire maggiore consapevolezza nelle fasi di progettazione successive.

***Tappa 2* - CREATIVITÀ SCIENTIFICA**
Si passa alla realizzazione pratica del progetto: bisogna mettersi in gioco per progettare, costruire e infine presentare agli altri le tappe del lavoro (processo) che sono state seguite. E alla fine... disassemblare tutti i pezzi e rimettere i materiali impiegati a disposizione di tutti!

***Dopo* - Riflettere, raccontare, condividere...**
Ripercorrere il percorso intrapreso e i processi messi in atto dai singoli studenti o dai diversi gruppi di lavoro. Insieme con gli studenti si potranno individuare parole-chiave o preparare disegni, brevi frasi o racconti. Infine, anche in questo caso potrete condividere la vostra esperienza di *tinkering* con noi scrivendo a **riconnessioni@fondazionescuola.it** e indicando il nome della scuola e la classe.

Kit C1 - *Ozobot*

Il kit C1 è proposto a tutte le classi della primaria, dalla prima alla quinta.



www.ozobot.com

***Ozobot* EVO** è un robot educativo, grande pochi centimetri, dotato di ruote e due micro-motori, 7 luci LED multicolore, un altoparlante, sensori ottici e sensori di prossimità. Si ricarica via cavo tramite porta USB e garantisce fino a 40 minuti circa di autonomia.

La sua peculiarità è quella di poter essere programmato senza la necessità di avere a disposizione un computer o un tablet. Il robot è infatti in grado di muoversi seguendo un percorso tracciato su un foglio con pennarelli: una pista, solitamente nera, intervallata da segmenti colorati, ciascuno costituito da una precisa sequenza di quadratini (rossi, verdi o blu) che corrisponde ad un diverso comportamento (cambio direzione, velocità di movimento, mosse speciali). La lista dei comandi predefiniti è scaricabile dal sito ufficiale, a questo link: https://goo.gl/bjcpK2. Sono disponibili online moltissime altre risorse, scaricabili gratuitamente, a disposizione dei docenti.

La facilità di programmazione grazie al codice colore consente di proporne l'uso anche per attività didattiche rivolte a studenti dei primi anni della scuola primaria e non richiede alcuna conoscenza pregressa di *coding* o robotica! Obiettivo principale: a partire dai comandi più semplici e da piste già preparate, familiarizzare con il tipo di "linguaggio" utilizzato e imparare poi ad elaborare sequenze di comandi articolate, che consentano al robot di muoversi all'interno dell'ambiente di lavoro attraverso una serie di mosse. Qualora il comportamento del robot si riveli diverso da quello atteso, sarà possibile procedere per tentativi per comprendere quale sia il modo più corretto per esprimere e comunicare la soluzione al problema posto sotto forma di semplici e chiare istruzioni.

Ad uso dell'insegnante, forniamo di seguito un breve elenco di corrispondenze tra i termini inglesi relativi ai comandi principali di *Ozobot* e le trasposizioni in italiano:

Lista comandi

SPEED / VELOCITÀ


Snail dose / andatura da lumaca (per 3 secondi)


Slow / lento (fino a nuovo comando)


Cruise / andatura standard, da crociera


Fast / veloce (fino a nuovo comando)


Turbo / molto veloce (fino a nuovo comando)


Nitro boost / andatura a razzo (per 3 secondi)

COOL MOVES / MOSSE SPECIALI


Zigzag / camminata a zig zag (procede ondeggiando alcune volte a destra e sinistra, poi riprende dritto)


Backwalk / camminata all'indietro (ruota di 180°, prosegue ondeggiando all'indietro per 1 secondo, poi ruota nuovamente di 180° e riprende dritto)


Spin / giravolta (ruota su sè stesso per due volte a velocità costante, poi prosegue dritto)


Tornado / tornado (ruota su sè stesso per due volte a velocità crescente, poi prosegue dritto)

DIRECTION / DIREZIONE


Go left / gira a sinistra (al prossimo incrocio)


Go straight / vai dritto (al prossimo incrocio)


Go right / gira a destra (al prossimo incrocio)


Line jump left / salta sulla linea a sinistra (ruota immediatamente di 90° a sinistra e avanza fino ad incontrare la pista successiva, poi prosegue su quella)


Line jump straight / salta sulla linea nera che incontra procedendo dritto (prosegue ad avanzare dritto anche oltre la fine della pista, fino ad incontrare la successiva - Attenzione! Il comando non ha valore se Ozobot incontra un incrocio prima di terminare la pista che sta percorrendo)


Line jump right / salta sulla linea a destra (ruota immediatamente di 90° a destra e avanza fino ad incontrare la pista successiva, poi prosegue su quella)


U turn / inversione a U (ruota di 180° e prosegue sulla stessa pista in direzione opposta)

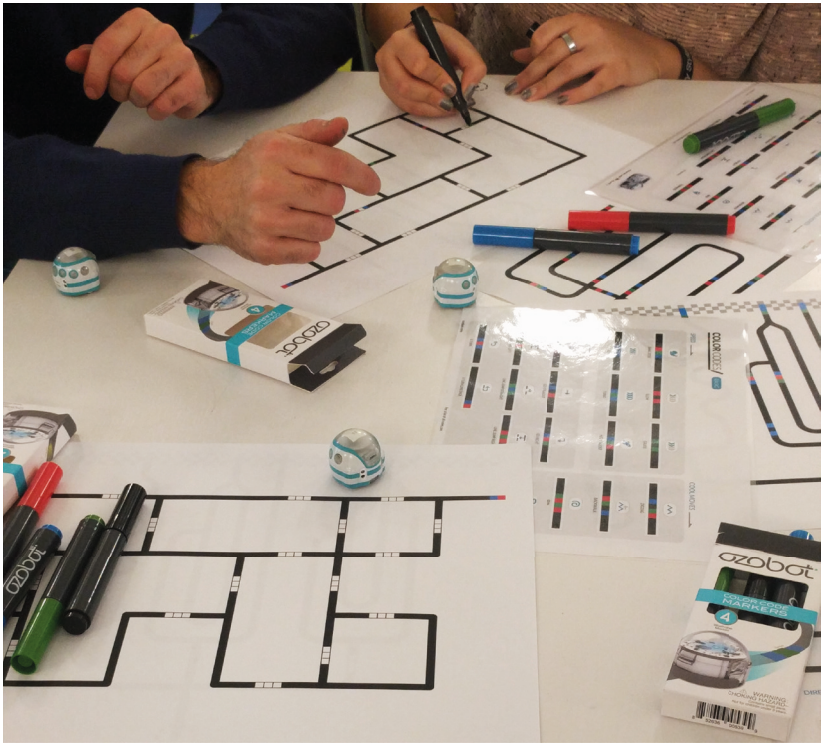

U turn (line end) / inversione a U (a fine corsa)



Quando si utilizza *Ozobot* per la prima volta è necessario eseguire una calibrazione: a tal fine, posizionarlo sopra un cerchio nero di area un po' maggiore della sua base, poi premere il tasto di accensione per 2-4 secondi e rilasciarlo quando il LED sulla sommità lampeggia con luce bianca. Attendere qualche istante (lasciando libero Ozobot di muoversi autonomamente) fintanto che la luce LED diventa verde: a quel punto la calibrazione è completata! Al contrario, nel caso in cui il LED si metta a lampeggiare in rosso, la procedura va riavviata.

Quando si disegna una nuova pista per Ozobot, è necessario ricordare alcune raccomandazioni (si veda la scheda disponibile al link: https://goo.gl/StC7fu):

- la **linea tracciata** non dev'essere troppo fine, né presentare tratti di spessore molto irregolare;
- è possibile inserire all'interno della pista delle svolte disegnando **tratti curvi** o **angoli retti** (sono invece da evitare gli angoli acuti);
- i **quadratini rossi, verdi e blu** che, posti in sequenza, costituiscono i comandi di cambio direzione, velocità o mosse speciali, devono essere tracciati giustapposti gli uni agli altri, senza sovrapposizioni né spazi bianchi intermedi e senza ricalcare il colore ripassando più volte con il pennarello. Gli stessi quadratini vanno posti unicamente sui tratti lineari della pista e non in corrispondenza di un incrocio. In caso di errori, è possibile sovrapporre e allineare al tratto della pista che si vuole modificare un'etichetta adesiva, su cui disegnare poi il comando corretto;
- tra i **comandi di direzione**, le istruzioni "*gira a sinistra*", "*gira a destra*" oppure "*vai dritto*" vanno inserite subito prima di un incrocio; in mancanza di uno di questi comandi, il robot segue di default la pista nera e/o decide in maniera arbitraria che direzione prendere una volta giunto ad un bivio.



Qualche spunto per l'attività in classe...

Per l'accensione di *Ozobot*, premere il tasto apposito e in seguito posizionare rapidamente il robot sulla pista, avendo cura di non coprire con le dita le due coppie di sensori di prossimità posti sul davanti e sul retro: questa procedura attiva la modalità di funzionamento standard, nella quale *Ozobot* segue la pista tracciata sul foglio.

Coprendo invece simultaneamente con le dita i quattro sensori di prossimità (fintanto che il LED anteriore non si illumina in verde), è possibile impostare una delle altre tre modalità di funzionamento del robot: *Follow* (*Ozobot* usa i sensori di prossimità anteriori per seguire il movimento del tuo dito); *Escape* (accelera e "scappa" quando si avvicina la mano ai sensori di prossimità posteriori); *Music* (produce note differenti a seconda del sensore di prossimità a cui si avvicina il dito).

L'attività in classe può essere avviata facendo lavorare gli studenti a piccoli gruppi, strutturando compiti via via più complessi. Di seguito viene proposto, a titolo esemplificativo, un possibile percorso, ma ciascun insegnante potrà ovviamente adattarlo alla propria classe o proporre agli studenti sfide diversificate:

1. osservare il comportamento di *Ozobot* ponendolo su una pista già preparata (e fornita nel kit) e provare a ipotizzare la corrispondenza tra ciascuna sequenza di quadratini colorati e il comando associato;
2. legenda alla mano (anch'essa fornita nel kit), far ripercorrere ad *Ozobot* la medesima pista e verificare di aver effettivamente compreso il codice colore e tutte le principali istruzioni;

3. davanti ad una sfida posta dall'insegnante (es. qual è la pista più rapida?) e impiegando un set di piste già pronte, identificare la strategia di risoluzione migliore oppure idearne una nuova;
4. utilizzando uno schema di pista già tracciata ma incompleta (nella quale le sequenze di quadratini sono già predisposte ma lasciate in bianco), inserire i comandi più adeguati a rispondere ad un obiettivo dato (es. uscire dal labirinto; far percorrere ad *Ozobot* tutti i punti della pista almeno una volta; oppure, al contrario, far percorrere tutta la pista senza che *Ozobot* torni mai sui propri passi, ecc.)
5. progettare e realizzare in autonomia una nuova pista.

In ciascuno di questi step viene di fatto posto un problema (*problem posing*) e si invitano gli studenti a identificare o costruire una soluzione (*problem solving*): la fase di verifica consentirà di confermare se il processo di lavoro messo in atto è stato efficace per giungere all'obiettivo o se invece è necessario apportare modifiche. In tale contesto, la codifica di istruzioni tramite un codice e la semplice programmazione diventano uno strumento per formulare razionalmente un problema e affrontarlo con un approccio logico-creativo.

Infine, si sottolinea che l'impiego di *Ozobot* e delle piste realizzate con il codice colore presentato a precedenza può essere integrato in progetti di tinkering più ampi. È possibile, ad esempio, simulare situazioni tratte dal mondo reale: immaginiamo di aver predisposto una pista che si inserisce in un "ambiente di lavoro" più articolato (es. presenza di ostacoli), all'interno del quale il robot dovrà essere in grado di muoversi e realizzare determinati compiti. *Ozobot* potrebbe, ad esempio, trovarsi davanti ad un segnale di stop (o un passaggio a livello, o la paletta di un vigile urbano...) azionato da un motorino, a sua volta messo in funzione grazie ad un semplice circuito elettrico (come quelli realizzati con il kit B). Se al suo passaggio *Ozobot* trova il segnale di stop abbassato davanti a sé, i sensori di prossimità anteriori gli imporranno di fermarsi; quando, per effetto della rotazione del motore, l'ostacolo sarà stato rimosso, il robot potrà invece riprendere la sua corsa.

Note aggiuntive per l'insegnante

Come mostrato durante l'incontro di formazione in presenza e riportato nei materiali messi a disposizione, è possibile non solamente realizzare piste con i pennarelli, ma anche crearle in formato digitale e poi stamparle in un secondo momento (e in più copie, se necessario). L'insegnante potrà facilmente raggiungere questo scopo grazie a software di uso comune, come Microsoft Excel o Word.
Inoltre, pur senza entrare nel dettaglio, si segnala che, avendo a disposizione un computer e una connessione internet, l'insegnante potrà anche sperimentarsi in un'altra modalità di programmazione di Ozobot, che consente di raggiungere esiti più articolati. A tal fine sarà sufficiente accedere ad OzoBlockly tramite il seguente link: https://ozoblockly.com/. Si tratta di un ambiente di programmazione visuale a blocchi, strutturato su cinque diversi livelli di complessità (Pre-reader, Beginner, Intermediate, Advanced, Master).

Kit C2 – Micro:bit

Il kit C2 è proposto alle classi terze, quarte e quinte della scuola primaria.



www.microbit.org

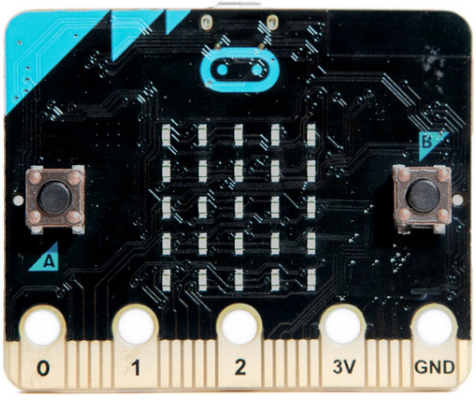
Micro:bit è un microcontrollore sviluppato dalla BBC in collaborazione con molti altri partner: nel 2016 è stata distribuita agli studenti inglesi e, a partire dall'anno successivo, resa disponibile anche in molti altri paesi.
Si tratta di una scheda programmabile di piccole dimensioni (4 cm x 5 cm) che può essere alimentata tramite collegamento con micro USB a computer o tablet oppure con alimentazione esterna di 3V, data da due batterie da 1,5V. Per la sua spiccata versatilità di utilizzo può essere impiegata in contesto scolastico per avvicinare al mondo del coding studenti più o meno giovani. Micro:bit può essere programmato con diversi linguaggi di programmazione e può comunicare con altri micro:bit o con dispositivi esterni.

Sulla scheda, oltre al microprocessore (il "cervello" di micro:bit) sono presenti:

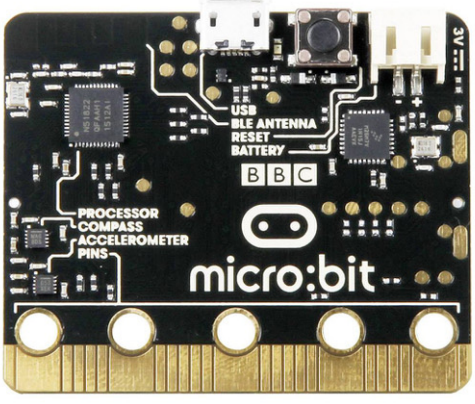
- sul fronte, 25 LED rossi (disposti su una griglia 5 x 5), che possono essere impostati su on/off e di cui è possibile controllare la luminosità. Per determinate applicazioni, seppure in maniera poco efficiente, possono essere impiegati anche come rilevatori di luce ambientale;
- sempre sul fronte, due pulsanti a pressione denominati A e B, che costituiscono dei dispositivi di input e che possono essere programmati in vario modo: la scheda è in grado di riconoscere quando viene premuto uno o entrambi i pulsanti e, di conseguenza, eseguire quanto previsto dal programma;
- i terminali GND (ground) e 3V per l'alimentazione;
- i pin numerati da 0 a 2 (e indicati nel seguito come P0, P1, P2) che costituiscono dei connettori di input-output.

Sul retro della scheda si trovano anche:

- un accelerometro, che rileva l'accelerazione sui tre assi e altri movimenti quali rotazione, scuotimento o caduta libera;
- una bussola elettronica, in grado di rilevare il campo magnetico terrestre (o altri campi magnetici) e riconoscere l'orientamento del micro:bit rispetto ad esso (direzione e gradi di rotazione);
- un modulo Bluetooth, che consente la comunicazione con altri micro:bit, PC, tablet o altri dispositivi (es. smartphone, fotocamere, ecc);
- un sensore di temperatura;
- un pulsante di reset.



Fronte



Retro

Qualche indicazione operativa...

Per programmare *micro:bit* è necessario connetterlo ad un computer (o tablet) e avere a disposizione una connessione internet.



Per la programmazione possono essere impiegati diversi linguaggi, ma in questo contesto prenderemo in considerazione solamente quello su cui è basato **Microsoft MakeCode Editor**: si tratta di una modalità di *programmazione grafica a blocchi*, cioè un tipo di programmazione visuale che consente di costruire un programma semplicemente spostando e ordinando in sequenza una serie di "blocchi" sul monitor del computer. Ad ogni "mattoncino" corrisponde un comando, che va incastrato con quello precedente e con il successivo: se due istruzioni o funzioni non sono tra loro compatibili, non risulterà possibile neanche incastrare i relativi blocchi. È certamente il metodo più semplice per avvicinarsi al *coding*!

Per accedere a MakeCode Editor bisogna collegarsi al sito web di *micro:bit*, cliccare su "*Let's code*" e poi selezionare "*New project*": si apre a quel punto la finestra dell'ambiente di programmazione, in cui è possibile, tra le impostazioni, selezionare la lingua italiana (https://makecode.microbit.org/).
Cliccando sull'icona con il punto interrogativo si può inoltre reperire la *Guida di riferimento* (o *Reference*), che fornisce una breve descrizione dei vari blocchi e del funzionamento dei comandi a cui corrispondono. Infine, dalla Home di MakeCode Editor è anche possibile visualizzare alcuni tutorial e lezioni (in lingua inglese) ed esempi di programmazione già realizzati che possono essere importati direttamente sul *micro:bit.*

Le informazioni per connettere la scheda al computer, accedere all'ambiente di programmazione e poi (come verrà spiegato in seguito) scaricare e installare il programma realizzato su *micro:bit*, sono raccolte anche nella *Quick Start Guide* (in inglese), reperibile al seguente link: https://microbit.org/guide/quick/.

La finestra di MakeCode Editor è suddivisa in tre parti: a destra, un'area di programmazione in cui vengono trascinati i blocchetti delle istruzioni; a sinistra, un simulatore che consente di visualizzare un'anteprima dei programmi realizzati prima di trasferirli concretamente sul *micro:bit* e i collegamenti di eventuali dispositivi esterni ai vari pin; al centro, un elenco delle "famiglie" di funzioni in cui sono organizzati i vari comandi.



Tra le voci di questo elenco troviamo:

- **Fondamentali** (*basic*): fornisce accesso alle funzionalità base del *Micro:bit* (es. all'avvio, per sempre, pausa, mostra icona, mostra LED, ecc);
- **Ingressi** (*input*): raccoglie i blocchi relativi a segnali di input e dati raccolti dai sensori (es. interruttore, sensore di luce ambientale; ecc);
- **Musica** (*music*): istruzioni relative alla generazione di toni musicali (es. tramite un *buzzer*);
- **LED** (*LED*): blocchi per il controllo della matrice di 25 LED integrati sulla scheda;

- **Radio** (*radio*): comunica dati utilizzando pacchetti radio;
- **Cicli** (*loops*): sequenze di istruzioni ripetute per un numero definito di volte o fintanto che una condizione resta verificata;
- **Logica** (*logic*): blocchi di istruzioni di controllo (es. del tipo *se – allora – altrimenti*) da eseguire alternativamente, se risultano soddisfatte alcune precondizioni;
- **Variabili** (*variables*): assegnazione o variazione del valore di una data variabile;
- **Matematica** (*math*): istruzioni legate all'esecuzione di semplici operazioni o funzioni matematiche (es. somma, differenza, radice quadrata, elevazione a potenza, ecc.);
- **Avanzati** (*advanced)*: raccoglie blocchi speciali molto diversificati (es. Funzioni, Array, Testo, Giochi, Immagini, Pin, Seriale, Controlli).

Cliccando su ciascuna "famiglia" di funzioni vengono visualizzati tutti i blocchi che ne fanno parte: resta con il puntatore del mouse per qualche istante sopra un blocco, ne viene visualizzata una brevissima descrizione. Una volta scelto il blocco di interesse, si trascina nell'area di programmazione a destra. Qualora invece si voglia eliminare un blocco, lo si trascina, al contrario, verso sinistra dall'area di programmazione verso l'elenco delle categorie di funzioni, eliminandolo.



Non si entra qui nel dettaglio dell'utilizzo di MakeCode Editor e degli esempi di programmazione, per i quali si rimanda al materiale presentato durante l'incontro di formazione in presenza e messo a disposizione degli insegnanti come risorsa.

Terminato un programma, per caricarlo sul *micro:bit* è necessario innanzitutto nominare e scaricare il file, che presenterà estensione *.hex*. Bisogna quindi connettere la scheda al computer tramite il cavo con micro USB (il *micro:bit* verrà visualizzato tra i dispositivi esterni collegati al computer) e, infine, sarà possibile recuperare il file del programma nel cartella *Download* e trascinarlo al suo interno (in maniera del tutto analoga a come si procederebbe per caricare un qualsiasi file su una chiavetta USB o pen drive).

Note per l'attività in classe



Come già anticipato, micro:bit può certamente essere impiegato per avvicinare gli studenti al coding, ma anche per integrare elementi di coding e robotica nelle attività di tinkering.
Accanto agli esempi di programmazione per i quali il micro:bit costituisce, al contempo, un dispositivo di input e output (come nel caso di molte delle esercitazioni svolte durante la formazione), si sottolinea la possibilità di interfacciare la scheda con interruttori e dispositivi elettrici di vario genere (come i componenti forniti nel kit B) o di far comunicare più micro:bit tra loro. Di seguito si fa cenno ad alcuni esempi applicativi, che, data la versatilità di applicazioni consentita da micro:bit, vogliono essere elementi di ispirazione, senza alcuna pretesa di essere esaustivi. Quindi... spazio alla creatività!
I dispositivi elettrici definiti a carico più leggero, come i LED o il buzzer possono essere collegati direttamente ai pin della scheda (pin P0 – P1 – P2), ad esempio per mezzo di cavi con morsetti a coccodrillo. In tal modo il micro:bit potrebbe essere impiegato per temporizzare l'accensione delle luci di un semaforo, oppure per produrre l'emissione di un suono ogni qual volta venga premuto un interruttore.

[A questo proposito, a titolo di esempio, si vedano tra i materiali forniti gli esercizi "suoni", "suoni con input", "semaforo" e "luci di Natale"].

Micro:bit può essere anche collegato ad un servomotore e programmato per azionare, ad esempio, il movimento di un tergicristallo o di un passaggio a livello. Volendo complicare ancora le cose, *micro:bit* potrebbe ad esempio, alla pressione di uno dei due pulsanti (A o B) che si trovano sul fronte, inviare un messaggio radio ad una seconda scheda, che a sua volta solleverà o abbasserà la sbarra del passaggio a livello.

[Si vedano, tra i materiali forniti, gli esercizi "tergicristallo", "passaggio a livello", "radio – passaggio a livello"].

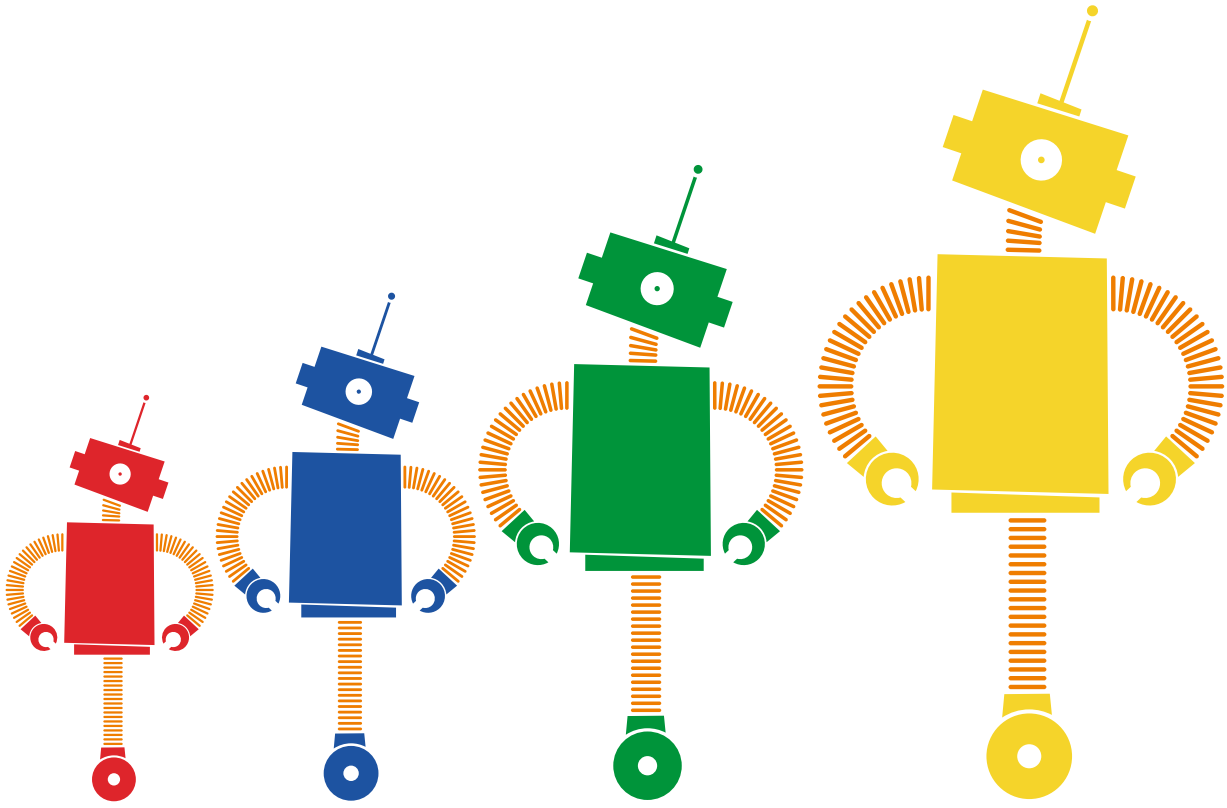


Tra i dispositivi elettrici forniti con il kit B, le lampadine a incandescenza, la ventola, i motorini e i motoriduttori richiedono per il loro funzionamento una corrente di intensità maggiore di quella erogata direttamente da *micro:bit*. Per questa ragione, tali dispositivi

non possono essere collegati direttamente ai pin della scheda, ma vanno interfacciati con un ***relay***, vale a dire un dispositivo che ha sostanzialmente la funzione di amplificatore di corrente.

[Si veda ad esempio l'esercizio "luci di casa", per il quale vengono collegate a micro:bit delle lampadine a incandescenza].

Ad ulteriore supporto per i collegamenti tra micro:bit e i dispositivi forniti con il kit B, si rimanda al breve video che accompagna questo testo. Buon lavoro e buon *computational tinkering*!



Restituiteci idee, progetti, foto, video e altro per condividere le vostre esperienze

riconnessioni@fondazionescuola.it

laboratoriocuriosita@xkezerotredici.it

Dalla laboratorietà al tinkering: per una didattica innovativa

ZeroTredici

Fondazione per la Scuola Compagnia di San Paolo

il LABORATORIO della CURIOSITÀ

RICONNESSIONI educazione al futuro